第64卷 第6期 天 文 学 报 Vol. 64 No. 6
2023年11月 ACTA ASTRONOMICA SINICA Nov., 2023

doi: 10.15940/j.cnki.0001-5245.2023.06.008

基于分布式执行框架的低频射电干涉阵列成像管线优化*

韦耀杰[1,2] 符杰林[1] 劳保强[2,3†]

(1 桂林电子科技大学信息与通信学院认知无线电与信息处理教育部重点实验室 桂林 541004)
(2 中国科学院上海天文台 上海 200030)
(3 云南大学物理与天文学院 昆明 650500)

摘要 平方公里阵列(Square Kilometre Array, SKA)项目是建设全球最大射电望远镜的国际合作项目, 其灵敏度和测量速度将比当前所有的射电望远镜都要高出一个数量级. 连续谱巡天是SKA的主要观测模式之一, 基于连续谱成像建立巡天区域的标准星图, 将能为后续天文科学研究奠定重要基础. 银河系与河外星系全天默奇森宽场阵列拓展巡天(GaLactic and Extragalactic All-sky Murchison Widefield Array survey eXtended, GLEAM-X)是2018—2020年利用SKA先导望远镜默奇森宽场阵列(Murchison Wide-field Array, MWA)二期拓展阵列开展的新的射电连续谱巡天项目, 观测期间积累了大量的低频巡天观测数据. 海量观测数据的自动化、大批量处理是SKA望远镜项目所面临的的最大挑战和难题之一, 基于分布式执行框架的成像管线优化经验将有助于解决海量数据处理问题. 详细介绍了GLEAM-X成像管线并对其进行整合和改进, 在中国SKA区域中心原型机(China SKA Regional Centre Prototype, CSRC-P)上实现了多条管线并行处理, 使用GLEAM-X观测数据验证成像管线系统的部署和测试其正确性. 随后为了优化管线提高处理效率, 使用数据激活流图形引擎(Data Activated Liu Graph Engine, DALiuGE), 将成像管线集成入DALiuGE执行框架中实现了管线的自动化分布式并行处理. 通过性能测试与结果分析表明, 基于DALiuGE执行框架进行优化的成像管线相较于传统的并行方式具有更好的性能优势、更灵活的适配性和可扩展性, 可支持未来SKA第一阶段试运行期间的大规模连续谱成像实验.

关键词 仪器: 干涉仪, 方法: 观测, 技术: 干涉测量, 技术: 图像处理, 射电连续谱: 普通
中图分类号: P164; **文献标识码**: A

1 引言

连续谱巡天是平方公里阵列(Square Kilometre Array, SKA)射电望远镜的主要观测模式之一, 主要涉及银河系和河外星系成像星表, 用于探查宇宙的恒星形成史[1]. SKA连续谱巡天观测是研究星系演化、宇宙中大规模结构的演化、宇宙磁场等关键科学目标的重要手段. 基于连续谱成像建立巡天区域的标准星图, 将能为后续天文科学研究建立重要科研基础.

SKA是一个国际合作大科学工程, 旨在建造世界上最大的射电望远镜[2]. 它将分两个阶段建造, 本文的成像管线主要为第一阶段的低频孔径阵列

2022-10-10收到原稿, 2022-12-02收到修改稿
*国家重点研发计划项目(2018YFA0404603)资助
†lbq19881213@gmail.com

SKA1-Low作先导应用. SKA1-Low将落于西澳大利亚的默奇森地区[3], 此外, 世界各地也已开展了多个SKA先导望远镜项目, 例如位于荷兰的低频阵列(Low Frequency Array, LOFAR)[4]、位于英国的增强型多元素远程链接干涉仪网络(enhanced Multi Element Remotely Linked Interferometer Network, e-MERLIN)[5]、位于澳大利亚的澳大利亚SKA探路者(Australian Square Kilometre Array Pathfinder, ASKAP)[6]和默奇森宽场阵列(Murchison Wide-field Array, MWA)[7–8].

位于西澳大利亚默奇森射电天文台的MWA望远镜是SKA三个先导望远镜之一. 银河系与河外星系全天默奇森宽场阵列巡天(GaLactic and Extragalactic All-sky MWA survey, GLEAM)是MWA 2013—2015年间的宽视场连续谱巡天, 在72–231 MHz的频率范围内对北纬30°以南的天空进行巡天覆盖[9]. GLEAM巡天为MWA留下了重要的低频巡天数据集, 并且GLEAM数据正在用于许多银河系、河外科学计划. 虽然GLEAM取得了巨大的科学成果, 其低频巡天数据有助于宇宙黎明和再电离时期的探测研究, 但它从根本上受到其较低分辨率和MWA原始配置的灵敏度限制, 仍有进一步优化和改进的空间.

银河系与河外星系全天默奇森宽场阵列拓展巡天(GaLactic and Extragalactic All-sky Murchison Widefield Array survey eXtended, GLEAM-X)是2018—2020年利用SKA先导望远镜MWA二期拓展阵列开展的新的射电连续谱巡天项目[10]. 该巡天的观测频率范围是72–231 MHz, 巡天覆盖范围与GLEAM巡天相同, 即北纬30°以南所有的天空区域, 约30000 deg^2. GLEAM-X的灵敏度和角分辨率能到达1–2 mJy和约45″, 分别是GLEAM的约6倍和2倍以上. 因此, GLEAM-X将能够探测到更多射电源, 所探测到的射电源也更清晰. 该巡天将能够产出包括连续谱与偏振图像、多频段星表、瞬变体搜索数据和电离层测量等科学数据产品, 其中连续谱图像和多频段星表是该巡天的首要科学数据产品. GLEAM-X巡天总共进行了超过40000次的快照观测, 总数据量约为2 PB, 处理完成所有观测数据需要近2000万CPU核小时. 面对GLEAM-X如此庞大的数据量, 急需开发自动化的并行处理管线进行数据的批量处理.

目前处理天文数据最常用的方法是在脚本中静态地定义工作管线的步骤组件, 这些脚本要么在本地机器上按顺序执行, 要么包装到作业脚本中, 提交给作业调度系统来执行[11]. 这对于SKA规模的数据处理(具有数千万个并发任务)并不可行, 而且管线的故障检测和后续恢复操作(例如重新执行)的代价非常高昂. 在数据并行处理方面, 工业数据密集型应用程序通常使用通用的数据并行框架, 如MapReduce[12]、Dryad[13]、Spark[14]等来处理大批量数据, 但直接使用它们来处理SKA天文数据时会出现两个问题: (1)大多数的数据并行框架需要将大的数据集拆分为小的数据块, 然后并行处理每个拆分块, 然而天文数据集通常涉及多个复杂维度来对数据集进行切片, 通用的数据并行框架难以支持SKA天文数据的多维度拆分; (2)现有数据并行框架的数据流优化对于商业工作管线可能非常有效, 但是它们在天文数据管线中的相关性和实用性却非常有限, 难以满足天文数据处理的科学标准.

此外, 现有的GLEAM-X管线还存在着一些问题: (1)管线的每个步骤都由独立的脚本文件来执行, 各个管线步骤的运行参数修改和部署实现较为繁琐, 不利于管线的自动化批量处理; (2)由于GLEAM-X的快照数据量较大, 成像时间更长, 单条管线的处理需要更多的计算资源, 且单条管线的处理时间较长, 需要一个能根据数据处理任务的计算需求自动分配硬件资源、同时处理大批量快照数据的高效稳定的多管线并行处理方法; (3)随着SKA建设的不断推进, 其所产生的数据量也将会增加数十倍, 传统的并行方式可能不足以支持如此大规模的数据流扩展并行, 因此, 并行处理方法还需要有更灵活的适配能力和更强大的可扩展性, 能将单条管线扩展为上万条管线并行处理.

因此, 针对以上问题, 我们将多个独立的管线步骤整合成规范统一的成像流程并验证其数据处理的正确性, 然后尝试使用基于任务排队系统的管线Bash脚本并行方式、基于Message Passing Interface (MPI)的并行扩展方式、基于Data Activated Liu Graph Engine (DALiuGE)执行框架的自动化

并行处理方式3种方法将原本的单一串行管线改进为多条管线并行的数据处理流程, 并通过对比各方法的综合运行性能来寻找最优的并行处理方案.

2 低频射电干涉阵列成像管线

本文的低频射电干涉阵列成像管线选取自MWA GLEAM-X成像管线, 本章主要介绍GLEAM-X观测数据、成像管线的详细流程步骤.

2.1 GLEAM-X观测数据

与GLEAM巡天相同, GLEAM-X整个天区巡天由7次不同位置的漂移扫描完成, 整个观测频段划分为5个子频段, 能提供接近连续覆盖的频率范围, 但弃用了受卫星等射电频率干扰严重的135–138 MHz附近的频段. 每个子频段的带宽为30.72 MHz, 每个子频段由连续的24个频率通道组成, 每个通道频率带宽为1.28 MHz, 每个频率通道标注为0–255的数字编号, 5个子频段中心频率通道编号分别为: 69、93、121、145和169. GLEAM-X观测是一系列为期4周共28晚的快照观测, 为了避免太阳光照对观测的影响, 漂移扫描观测仅在夜间进行. 在一晚的观测时间内, 每个子频段以0.5 s的时间分辨率每120 s记录一次数据, 称为2 min快照观测数据, 10 min则完成5个子频段数据的观测. 每个快照数据文件以起始观测的GPS时间作为ID号命名.

2.2 成像管线流程步骤

本文的低频射电干涉阵列成像管线是基于开源的GLEAM-X成像管线开发[1], 管线主要用于处理GLEAM-X的快照观测数据. 该管线的主要流程步骤如图1所示, 具体步骤描述如下:

(1)原始快照观测数据下载(Download Data). 目前, 快照数据主要通过MWA全天虚拟天文台(All-Sky Virtual Observatory, ASVO)[15]下载, MWA ASVO给用户提供了两种下载方式. 第一种方式是利用ASVO网页交互式界面的'New Data Job'功能提交下载作业, 等待作业完成后, 通过提供的下载链接手动下载数据. 第二种方式是, 利用ASVO提供的命令行客户端(manta-ray-client), 将需要下载的快照数据的观测ID写入逗号分隔值(Comma-Separated Values, CSV)格式文件, 提交下载作业后, 可以实时监测下载作业的状态并在作业完成后自动下载数据. 因此, 第二种方式比较适用于数据的大批量自动下载. 下载的原始观测快照数据为ZIP压缩文件, 该压缩文件包含: 可见度数据文件(gpubox.fits)、观测元数据文件(metafits和ppds)、射电频率干扰(Radio Frequency Interference, RFI)初步标记文件(flags.zip和mwaf);

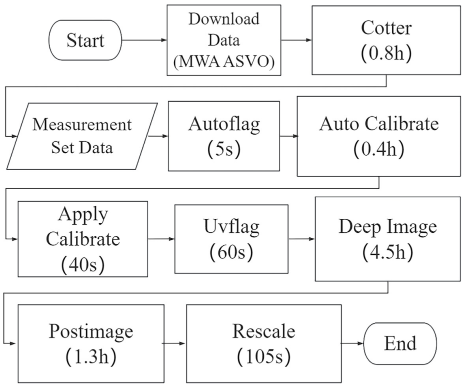

图 1　GLEAM-X快照数据成像管线

Fig. 1　Imaging pipeline for GLEAM-X snapshot data

(2)快照数据解压与数据预处理(Cotter). 对下载完成的快照数据压缩文件进行解压, 然后进行数据预处理. 数据预处理步骤采用的是Offringa等人开发的Cotter软件[16], 该软件是MWA数据专用的数据预处理软件, 通过读取可见度、观测时间与频率等数据, 按照预设的4 s时间分辨率和40 kHz频率分辨率对可见度数据进行时间平均和频率平均; 之后读取RFI初步标记文件信息, Cotter使用André Offringa's Flagger (AOFlagger)[17]软件中的算法进行RFI检测和损坏天线的标记, 并计算相关统计数据和校正电缆长度延迟相位; 最后, 由于后续步骤的数据处理软件或程序是基于CASA核心软件库Casacore[18]开发的, 还需要将处理后的文件转换为通用天文软件应用程序Common Astronomy Software Applications (CASA)的MeasurementSet (MS)[19]格式文件;

[1] https://github.com/tjgalvin/GLEAM-X-pipeline

(3)故障天线标记(Autoflag). 由于步骤(2)中的损坏天线标记是根据观测时的记录文件进行的, 个别天线故障没有被及时记录, 这一步骤是根据观测工程师或者校准源数据处理中提供的天线受损记录, 进行重新标记, 以免有漏标的故障天线影响后续的数据处理结果;

(4)天空模型建立与校准解生成(Auto Calibrate). 鉴于GLEAM-X与GLEAM的巡天天区覆盖范围和观测频率均相同, 因此可以使用现有的GLEAM视场模型对GLEAM-X观测数据进行校准. 首先, 在已发表的GLEAM星表中寻找GLEAM-X快照观测数据视场范围内同波段、同位置的射电分量或源信息(流量密度和位置等). 然后, 基于这些信息制作出该快照数据的初始天空模型, 并按照Andre格式输出模型文件[2]. 最后, 天空模型文件和快照数据输入到校准算法进行校准解的产生, 校准算法采用Offringa等人提出的MitchCal算法[20], 该算法将计算出天空模型与实际快照观测的可见度数据所有点的幅度和相位偏移量, 这些偏移量再进行最小二乘法拟合获得最终的校准解, 这个校准解可以修正实际快照数据的幅度和相位误差;

(5)目标场数据校准(Apply Calibrate). 将步骤(4)中生成的校准解应用于快照观测的可见度数据进行幅度和相位校准, 这一步能够消除观测设备和大气等大部分干扰因素的影响;

(6)基线数据标记(Uvflag). 扫描检查校准后的可见度数据是否有遗漏的RFI未进行标记, 将标记的故障天线对应UV基线的可见度数据进行标记. 该步骤能够尽可能地消除广播调频信号和数字电视信号对该频段观测数据的干扰影响;

(7)深度成像(Deep Image). 深度成像主要利用w方向叠片洁化(W-Stacking Clean, WSClean)软件[21]中的$wsclean$命令实现, 最终生成2 min快照图像. 主要的成像参数设置如表1所示. 在成像方面, GLEAM-X的观测视场大小与GLEAM相同, 但角分辨率提高了约2倍, 因此输出的图像大小需增大2倍, 即8000×8000. 每个像素的大小(像素分辨率)一般设为望远镜角分辨率的四分之一或五分之一, 可以由基础像素分辨率除以子频段中心频率的编号数(单位rad)得到每个数据的具体像素分辨率, 其中基础像素分辨率为0.6. 由于GLEAM-X观测视场较大(约$30° \times 30°$), 视场范围内将包含不同大小尺度的射电源, 传统的洁化(Clean)方法已经无法满足其成像处理的更高动态范围的要求, 而WSClean支持多尺度洁化(Multi-scale Clean)算法[22], 能够更好地重建延展结构的射电源. 因此在参数设置上选择multiscale的主要迭代洁化增益为0.85, 即在每个主要迭代中减去85%的流量密度, 多尺度增益参数使用默认值0.15. 最初的GLEAM数据使用了具有鲁棒性参数为-1的"Briggs"图像加权[23], 但这种加权不适用于MWA二期扩展配置, 对于GLEAM-X, 通常首选natural的加权模式以最大限度地提高灵敏度, 但是与GLEAM相比, natural加权的角分辨率提高仅为1.5倍, 并且点源灵敏度没有最大化. 为了在保持整体灵敏度的同时平衡分辨率的提高, 最终选择了+0.5鲁棒性参数的"Briggs"图像加权[24], 它提供的自动调节权重(weighting)方法, 可以获得更均衡的成像效果. 此外设置洁化最大迭代次数为10000000次, 迭代阈值在均方根RMS (Root Mean Square)噪声$1\sigma - 3\sigma$之间动态调节, 在成像中所选取的数据为校准过后的CORRECTED_DA-TA数据列;

(8)成像后处理(Postimage). 对深度成像生成的图像执行电离层校正, 主要为了消除电离层引起的干扰. 电离层的干扰会导致射电源的相位偏移, 该偏移会随电离层位置的不同而发生变化. 对于成像后的快照图像, 首先使用背景噪声估计(Background And Noise Estimation, BANE)[25]工具计算背景和RMS噪声σ, 并使用源查找软件包Aegean[25]进行源查找, 最小阈值为5σ. 之后根据Hurley-Walker等[26]的方法, 使用fits_warp[27]软件包的星表交叉匹配功能, 将查找到的源与GLEAM巡天星表进行交叉匹配, 通常保留约3000个交叉匹配源, 从中选取750个较亮的源, 更多的源无法提高校正的准确性, 而且会增加计算负担, 因此选择该值作为收益递减点. 然后, 根据快照图像中保留的源与参考星表源之间的位置差异计算偏移量, fits_warp使用这些偏移量来创建修正模型, 将

[2]https://github.com/PaulHancock/MWA-SkyModel

其应用于原始快照图像, 并将修正后的结果内插回图像完成位置校正. 之后使用flux_warp[28]软件包以类似的方式进行流量密度校正, 最后重新使用Aegean对处理后的快照图像进行源查找, 对比源的数量检验校正效果;

表 1 wsclean成像主要参数设置

Table 1 Main parameter settings of wsclean imaging

Parameter	Setting
-size	8000 × 8000 pixels
-scale	Base pixel resolution / sub-band center frequency channel number rad
-weight	'briggs' Mode, 'robust' 0.5
-multiscale-mgain	0.85
-multiscale-gain	0.15
-niter	10000000
-auto-mask	3σ
-auto-threshold	1σ
-data-column	CORRECTED_DATA

(9)图像重设(Rescale). 对步骤(8)处理后的图像进行图像重缩放. 首先, 读取上一步Aegean对图像进行源查找的结果, 使用stilts软件[29]将其与步骤(4)的天空模型进行交叉匹配, 获取快照图像与参考模型的赤经(Right Ascension, RA)和赤纬(Declination, Dec)的偏移量以及流量密度的比值, 使用sigma_clip算法[29]来识别和去除小于中位数减标准差或大于中位数加标准差的异常值数据, 接着对修正前后的RA和Dec数据进行多项式拟合并绘制拟合曲线的图像以供后续的误差分析使用, 然后生成快照图像的背景、RMS和拼接权重文件, 最后根据RA、Dec的偏移量(Δ_{RA}、Δ_{DEC})计算出GLEAM-X快照图像的矢量化函数, 对图像像素进行矢量化处理, 重设图像的比例尺, 生成重新缩放的fits文件.

3 基于任务排队系统的成像管线Bash脚本并行优化

本章主要介绍在CSRC-P (China SKA Regional Centre Prototype)原型机上对管线的部署实现和并行优化, 并对其正确性进行验证.

3.1 成像管线在CSRC-P上的环境部署

开源版本的GLEAM-X管线是基于Singularly容器镜像环境和澳大利亚Pawsey超算中心Galaxy超级计算机软件环境开发和运行的[30]. 为了能够在CSRC-P原型机上部署并运行该成像管线, 我们进行了管线软件环境的部署以及相关代码的修改.

所部署的软件包括: cotter v4.6、AOFlagger v3.0、mwa-reduce-2022、WSClean v2.9、stilts v3.4等. 由于原型机所有节点安装的操作系统为Centos 7.0[3], 我们编写了一个能自动化进行所有软件部署安装的Bash脚本, 脚本内包含了下载、解压、编译、配置、安装等软件部署步骤的命令语句, 且涵盖了需要以同样步骤安装的大量前置依赖软件包的部署命令. 为了便于用户动态地修改自己的软件环境, 且不影响其他用户的软件环境, 实验的软件环境采用环境变量管理工具Modules[4]进行管理, 当进行GLEAM-X巡天数据处理时, 只需要在提交的任务脚本中载入对应软件的模块文件modulefiles, 即可使用所部署的数据处理软件环境.

此外, 通过多次实验发现, 由于GLEAM-X的快照数据量较大、成像时间更长, 与GLEAM数据的一个计算节点可以同时运行多条管线的处理方式不同, GLEAM-X需要更多的内存和计算资源, 单个快照数据处理就需独占一个计算节点. 例如, 使用两个计算节点并行处理两个GLEAM-X快照数据包平均总用时7.2 h, 而使用单个计算节点处理相同的两个快照数据包平均总用时9.5 h, 处理时间同比增加31%. 其主要原因在于, 在深度成像中WSClean采用的大视场成像算法是w方向叠片(w-stacking), 其中w是基线(u、v、w)坐标系的w轴, 从地心指向目标源相位中心方向. 该算法首先将

[3] https://www.centos.org/

[4] http://modules.sourceforge.net

三维可见度数据根据w值, 划分为不同w层的二维可见度数据, 然后分别对不同w层的二维可见度数据进行栅格化和FFT (Fast Fourier Transformation)成像, 最后不同w层得到的图像乘以w因子并进行叠加得到最终的图像结果[21]. 消耗的内存和计算资源与w的大小和图像的像素大小有关. 由于需要产生的快照图像的像素大小增大了, 所以在深度成像时, 消耗的内存和计算资源也大大增加, 对于具有大规模计算节点资源的SKA区域中心(SKA Regional Centre, SRC)来说, 能够大大缩短数据处理时间, 更适合于将来大规模连续谱巡天数据的处理. w-stacking的时间复杂度(Time Complexity, TC)[21]由下式给出:

$$\mathrm{TC}_{w-\mathrm{stacking}} = O(N_{\mathrm{pix}}^2 \log_2(N_{\mathrm{pix}} w_{\max} \sin\alpha_{\mathrm{FOV}} + N_{\mathrm{vis}}))\,, \quad (1)$$

其中, O表示算法的最坏时间复杂度, 即算法在最坏情况下所需的时间复杂度, N_{pix}为图像像素大小, N_{vis}为可见度数据长度, $w_{\max}$为最大w值, α_{FOV}为成像视场(Field Of View, FOV).

3.2 成像管线在CSRC-P上的任务排队并行优化

GLEAM-X成像管线所有处理步骤的执行命令和代码均采用Bash脚本进行封装[31], 为了提高数据处理效率, 保证单个计算节点处理单条管线, 我们设计了批量数据处理方法, 即通过任务排队系统同时并行处理多个快照观测数据. 图2表示基于Bash脚本任务排队并行优化的基本原理, 具体如下:

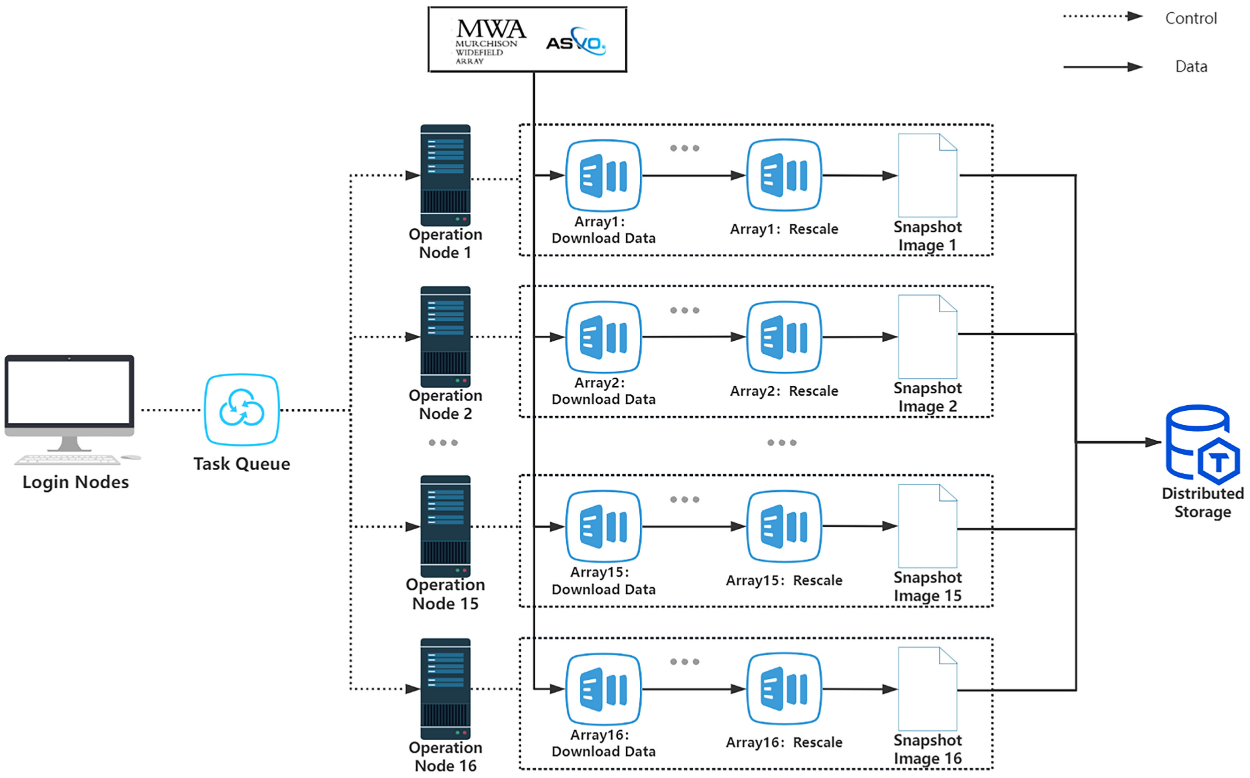

图 2 基于Bash脚本任务排队并行优化原理图

Fig. 2 Principle diagram of task queuing and parallel optimization based on Bash script

(1)通过充分利用原型机作业调度系统中的任务依赖和计算节点分配功能进行并行处理, 我们编写了一个总管线任务脚本, 其中涵盖了管线所有处理步骤的子脚本, 并将每个步骤所需的运行参数做了顶层设置, 可以根据实际运行环境在总任务脚本顶部进行统一设置和修改;

(2)用户在登陆节点(Login Nodes)提交总任务脚本, 通过计算节点分配功能将整条管线步骤队列(Array)的每个子步骤分配到同一个计算节点(Operation Node)上运行, 利用前一个步骤提交后返回的任务编号(jobid)和任务依赖来实现任务排队, 待上一个步骤完成后自动执行下一个步骤任务, 实现单条管线的处理运行;

(3)并行运行多条管线时, 在单条管线的最外层, 将单条管线的执行放入到循环结构中, 通过定义任务队列(Task Queue)的循环次数(即并行运行的管线条数)在单个循环内确定单条管线的标识ID和分配的计算节点编号. 将每个步骤的脚本文件提交到对应管线的计算节点上进行任务排队, 通过多次循环地提交任务来将多条管线分配到多个计算节点(即一个计算节点执行一条管线处理), 实现基于任务排队的管线Bash脚本的多节点并行处理.

3.3　并行成像管线的GLEAM-X数据验证

为了验证部署的环境以及修改后的管线的正确性, 我们选取一个GLEAM-X快照原始观测数据进行测试. 快照数据观测ID是1212314512, 观测起始时间为协调世界时(Universal Time Coordinated, UTC) 2018-06-06 10:01:34, 观测相位中心为(RA, Dec) = (176.87°, −25.78°). 管线处理该快照数据用时6.7 h, 最终生成图像数据包含多频率综合(Multi Frequency Synthesis, MFS)和4个子频段的总强度即射电偏振斯托克斯I分量(Stokes I)图像结果, 频率范围为170–200 MHz, 图像RMS噪声4.2 mJy/beam与GLEAM-X文献[10]给出的结果相符.

为了进行对比分析, 选取MFS的总强度图像进行射电源搜寻并建立星表, 使用Aegean软件搜寻出的射电源共有11409个, 使用星表和表格操作工具(Tool for OPerations on Catalogues And Tables, TOPCAT)[32]进行交叉匹配, 设置允许的最大偏差为45 arcsec (GLEAM-X角分辨率). 与已发表的GLEAM星表匹配的源个数为10234, 匹配的源个数占源总数的89.7%, 剩余匹配的主要原因是GLEAM-X的分辨率和灵敏度较高, 相较于GLEAM能够探测到更多的射电源, 因此新多出的射电源与原有的GLEAM星表存在不匹配现象. 与广域红外巡天探测器(All the Wide-field Infrared Survey Explore, ALLWISE)星表匹配的源个数为11399, 匹配的源个数占源总数的99.9%, 表明获得的射电星表几乎都能够找到红外波段的对应体. 通过统计指标$\Delta_{\rm RA}$、$\Delta_{\rm DEC}$和Peak_flux的比值得到如下统计图3和表2. 图3为GLEAM-X快照图像星表分别与ALLWISE星表和GLEAM星表交叉匹配的统计分布图, 并绘制了分布图的拟合曲线, 表2为图3中各个分布函数的高斯拟合结果, 分别统计了各个分布函数高斯拟合的平均值、幅度值、标准差. 通过与两个星表进行匹配, $\Delta_{\rm RA}$、$\Delta_{\rm DEC}$的均值均在0左右, Peak_flux比值的均值在1左右, 标准差在0左右, 标准差能反映一个数据集的离散程度, 标准差越小则数据离散越小, 由统计结果可以得出经由我们改进过后的GLEAM-X成像管线在CSRC-P上的数据处理结果是可靠的.

表 2　图3中各个分布函数的高斯拟合结果
Table 2　Gaussian fitting results for each distribution in Fig. 3

Distribution label	Average value	Amplitude	Standard deviation
(a)	−0.3 arcsec	1258.4	5.9 arcsec
(b)	−0.9 arcsec	1221.6	5.3 arcsec
(c)	1.02	1123.5	0.2
(d)	0.04 arcsec	1676.2	4.3 arcsec
(e)	−0.03 arcsec	1704.3	3.4 arcsec

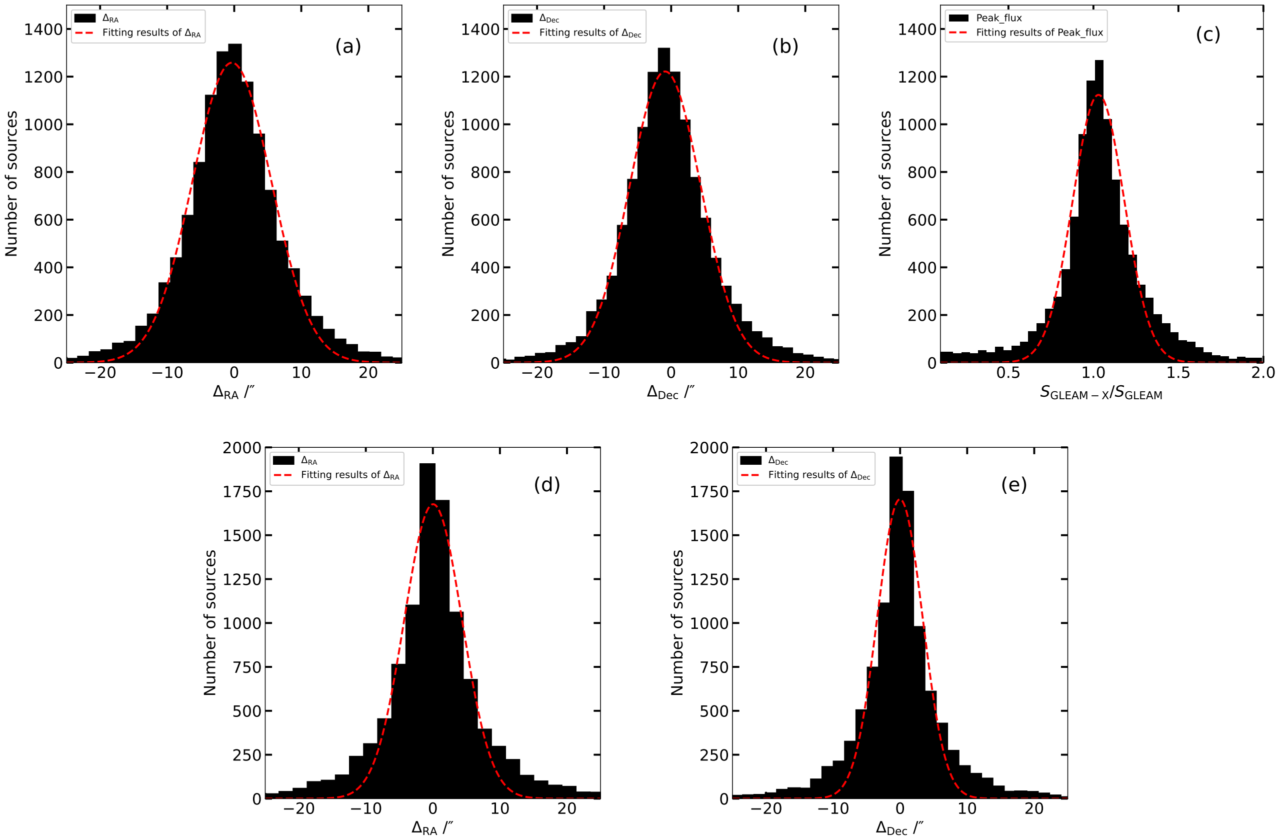

图 3 图(a)和(b)为GLEAM-X快照图像星表与ALLWISE星表交叉匹配统计结果. 图(a): $\Delta_{\rm RA}$分布图; 图(b): $\Delta_{\rm DEC}$分布图. 图(c)–(e)为GLEAM-X快照图像星表与GLEAM星表交叉匹配统计结果. 图(c): Peak_flux比值分布图, $S_{\rm GLEAM-X}$为GLEAM-X星表的Peak_flux值, $S_{\rm GLEAM}$为GLEAM星表的Peak_flux值; 图(d): $\Delta_{\rm RA}$分布图; 图(e): $\Delta_{\rm DEC}$分布图. 图中红色虚线为分布图的拟合曲线.

Fig. 3 Panels (a) and (b) display the statistical results of cross-matching between the GLEAM-X snapshot image catalog and the ALLWISE catalog, respectively. Panel (a): distribution of $\Delta_{\rm RA}$, $\Delta_{\rm RA}$ is RA difference of two catalogs; Panel (b): distribution of $\Delta_{\rm Dec}$, $\Delta_{\rm DEC}$ is DEC difference of two catalogs. Panels (c) to (e) show the statistical results of cross-matching between the GLEAM-X snapshot image catalog and the GLEAM catalog. Panel (c): Peak_flux ratio distribution, $S_{\rm GLEAM-X}$ is Peak_flux of the GLEAM-X catalog, $S_{\rm GLEAM}$ is Peak_flux of the GLEAM catalog; Panel (d): distribution of $\Delta_{\rm RA}$; Panel (e): distribution of $\Delta_{\rm Dec}$. Red dashed lines are the fitting curve of the difference distributions.

4 基于MPI的管线脚本并行扩展

为了更全面地开展性能测试对比, 我们对原型机上部署的基于Bash脚本的管线进行了MPI扩展修改, 以此来与其他的扩展方式进行性能比较.

MPI是一个跨语言的通讯协议, 属于消息传递式并行程序设计, 在今天仍为高性能计算的主要模型[33]. 问题的分解策略、进程间的数据交换策略都需要由用户来手动确定, 在挖掘潜在并行性方面更主动, 但并不利于流程的自动化处理.

为了实现MPI并行扩展, 我们使用python编程获取通信域MPI_COMM_WORLD中的进程号, 通过python内置函数os.system对原本管线步骤的每一个Bash脚本生成子进程执行, 并以MPI函数的形式来进行调用. 在原本所需的脚本参数的基础上新增一个进程ID号, 用以识别划分多条管线并行运行且独立计算每条管线的运行时间. 之后再额外提交一个设置和运行MPI的任务脚本来执行MPI扩展程序, 通过修改预分配的节点个数来设定并行运行的管线条数, 最终能够获得与其他并行方式相同的成像结果.

5 基于DALiuGE执行框架的成像管线自动化并行实现

本节主要介绍DALiuGE执行框架的处理机制以及将成像管线移植到执行框架中的两个主要实现部分: 管线步骤的DALiuGE执行框架组件开发; 管线的DALiuGE逻辑图组合和物理图部署运行.

5.1 DALiuGE处理机制

数据激活流(Liu)图形引擎(DALiuGE)是一个专门用于以图形方式处理超大规模射电天文数据集的工作流图形执行框架[11]. DALiuGE提供分布式数据管理平台和可扩展的管线执行环境, 以支持射电天文数据的自动化、软实时、数据密集型处理.

与现有的处理框架相比, DALiuGE的主要优势在于:

(1)自动化资源调度管理: DALiuGE将管线的逻辑图与其运行时的物理图实现分离, 管线的运行能够以负载平衡、数据移动最小化成本的方式映射到当前可用的计算资源上, 能够均衡整体工作负荷(包括计算时间和内存使用)以实现最佳的资源调度分配;

(2)数据驱动: DALiuGE的执行是由数据激活的, 每个单独的数据项都会自动触发对其自身的处理流程. DALiuGE将数据建模为图的节点(Drop), 将它们视作为可管理的实体, 允许数据和应用程序触发和接收事件. 这种去中心化也使得执行框架具有很强的可扩展性和灵活性, 由于其完全去中心化的执行模式, DALiuGE可以并发管理和执行数千万个任务;

(3)生命周期管理: DALiuGE还在执行引擎中集成了一个数据生命周期管理组件, 能够跟踪Drop的执行并在必要时自动迁移或删除[11].

DALiuGE的开发很大程度上是基于射电天文的处理需求, 并且DALiuGE采用了通用的、数据驱动的框架架构, 适用于许多其他数据密集型应用程序, 因此我们选择以DALiuGE执行框架来实现GLEAM-X成像管线分布式并行优化.

5.2 管线步骤的DALiuGE执行框架组件开发

由于GLEAM-X管线所有处理步骤的执行命令和代码均采用Bash脚本进行封装, 而DALiuGE的Drop组件主要是使用python语言编写. 虽然DALiuGE自带有简单的Bash脚本组件可以使用, 但直接使用Bash脚本组件执行效率不高. 从本质上来说, 这种移植方式并不能充分发挥DALiuGE的性能, 因此我们选择将GLEAM-X管线的整体代码移植到DALiuGE的python组件中, 使其成为真正的DALiuGE组件程序. 无论使用哪种方法, 我们都将确保最终处理结果与原GLEAM-X管线一致.

DALiuGE的核心是数据驱动. Drop是一个无状态程序单元, 当Drop接收到数据时开始运行处理, 并在处理完成时输出数据. 我们在对GLEAM-X管线代码进行详细分析的基础上, 将每一个管线步骤重新编写为DALiuGE组件, 构建了一系列Drop组件. 这些Drop组件继承自DALiuGE的两个标准Drop, 即PythonApp和FileDrop. PythonApp用于执行脚本程序, 需要自行编写, FileDrop用于传输变量, 在连接管线流程图时根据输入输出的端口自动生成.

关于组件的编写, DALiuGE提供一个组件项目模板[5], 该模板涵盖应用程序组件和数据组件, 包括项目设置、测试、格式合规性、构建、文档、发布和持续集成等. 组件的编程需要修改apps.py文件来实现, 通过组件规定的注释方式可将App所需的参数在天文图形语言环境编辑器(Editor for the Astronomical Graph Language Environment, EAGLE)可视化为可修改选项, 方便用户随时修改而不再需要涉及py文件的编程. 此外, 可以为组件添加自定义的输入和输出端口, 端口是获取数据和信息进出组件的主要方式, 端口始终连接到数据组件, 并为应用程序组件提供同质I/O接口, 默认情况下只允许匹配端口之间的连接, 可以将任何想要的内容写入输出端口, 但需要注意其他组件要能够正确理解和解释.

图4为DALiuGE中成像管线的逻辑图, 包含PythonApp和FileDrop共20个Drop, 其中执行主

[5] https://github.com/ICRAR/daliuge-component-template

要功能的PythonApp有10个, 分别是1个为了满足DALiuGE中Scatter组件的并行需要而编写的并行配置App Drop (ProduceConfigApp)、根据管线主要步骤编写的9个App Drop和连接管线流程图时根据输入输出端口自动生成用于传输任务编号(jobid)的10个FileDrop. 具体说明如下:

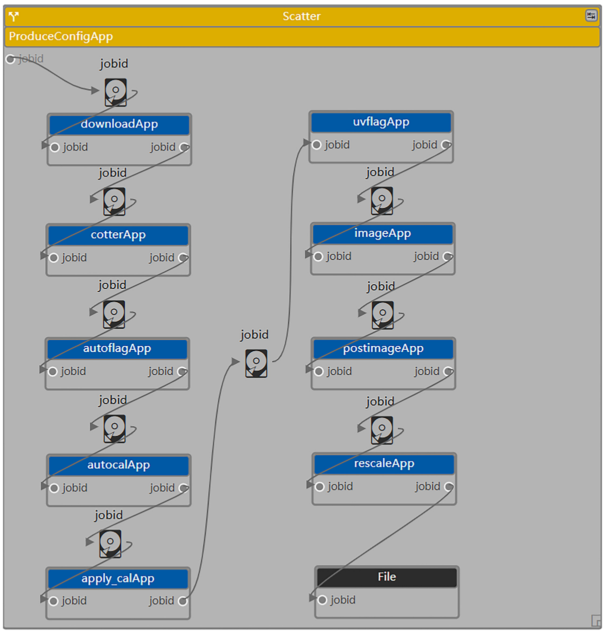

图 4 成像管线的逻辑图

Fig. 4 Logic graph for the imaging pipeline

ProduceConfigApp: 并行配置. DALiuGE自带的Scatter组件会根据组件参数(Number of copies)的数目将包含在内的管线流程拆分为多个并行处理实例, 因此需要根据Scatter的输出数量, 按照总观测ID下载CSV文件和包含全观测ID的文本文件, 通过ProduceConfigApp为每一条并行处理管线配置相应的单条下载CSV文件和观测ID文本文件, 并将CSV文件名和文本文件名传输到下一个Drop.

9个App与成像管线中的步骤按顺序对应. downloadApp: 数据下载; cotterApp: 数据预处理; autoflagApp: 故障天线标记; autocalApp: 生成校准解; apply_calApp: 目标场数据校准; uvflagApp: 基线数据标记; imageApp: 深度成像; postimageApp: 成像后处理; rescaleApp: 图像重设.

5.3 管线的DALiuGE逻辑图组合和物理图部署运行

以上的PythonApp组件编写完成后为py文件, 还需要使用DALiuGE中的转换文件生成对应组件的palette文件. 在EAGLE[11]中导入组件palette文件, 便可在EAGLE中生成可使用的组件模块. 拖放组件模块将单条管线所需的组件按照管线步骤依序连接并包含在Scatter组件范围中, 通过改变Scatter中的'Number of copies'参数来选择将管线并行为想要的条数, 最终组合成管线逻辑图.

管线逻辑图搭建好后, EAGLE提供Translation选项将逻辑图转化为物理图, 如图5所示. 物理图由DALiuGE翻译引擎从逻辑图计算而来, 然后显示在EAGLE界面中. 它表示将逻辑图转换为管线的物理执行图, 并将该图映射到实际的计算机集群上.

当Translation过程产生的物理图在DALiuGE执行引擎上部署和实例化时, 就会成为跨多个资源单元的分布式执行计划中相互连接的Drop集合[11]. 由于该管线需要在CSRC-P原型机上以分布式多节点方式并行运行, 因此需要在原型机的DALiuGE中进行部署执行. 这需要将之前组合好的管线逻辑图保存为graph文件存储入原型机系统内, 通过提交多节点任务脚本将graph文件以DALiuGE的集群方式部署执行. DALiuGE会根据给定的计算节点数自动调度并行管线的运行, 在计算节点与并行条数相当的情况下, 均匀分配实现单个计算节点运行一条成像管线的最佳方式运行. 同时在提交目录下产生节点管理的log文件、运行空间的环境文件、监测运行状态的输出文件, 可以实时地了解和监控执行进度. 待最后一条管线执行完毕后DALiuGE会自动结束任务, 最终实现自动化的分布式并行成像管线处理.

6 性能测试与结果分析

6.1 并行方式总结

通过对上述并行方式的实现, 我们总结了3种并行方法的优化特性:

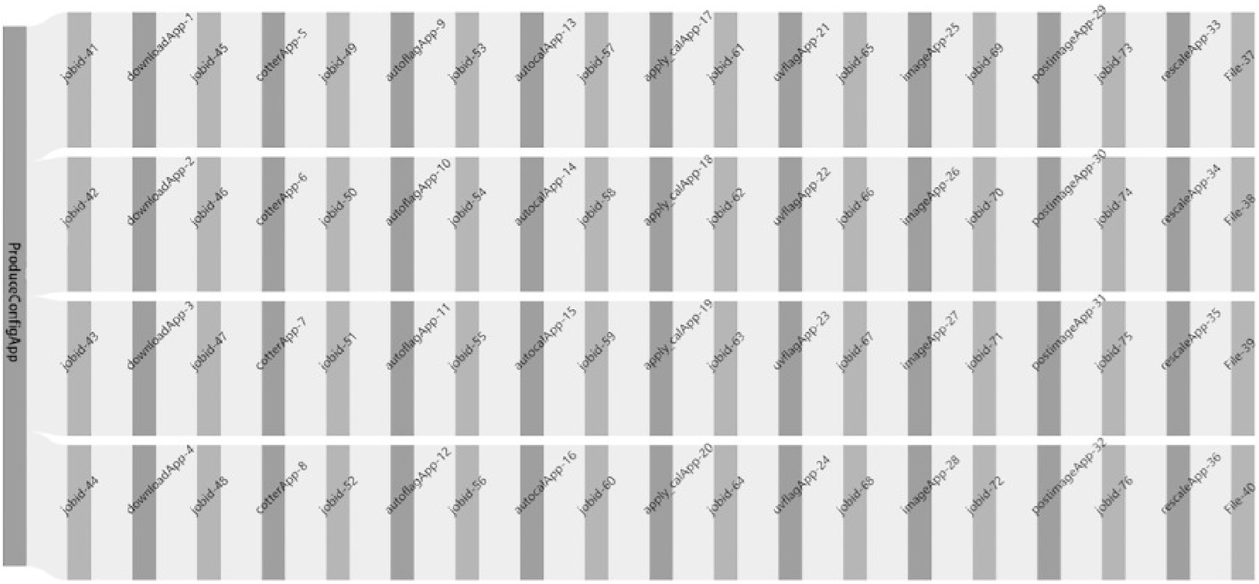

图 5　成像管线的物理图. 图为将Scatter内的Drops拷贝为4份, 然后并行处理4个GLEAM-X数据的物理图.

Fig. 5　Physical graph for the imaging pipeline. The panel represents the physical graph of the imaging pipeline, where the Drops in Scatter have been replicated into 4 copies to enable parallel processing of 4 GLEAM-X data.

(1)在资源调度和任务分配方面, 基于任务排队的Bash脚本并行和MPI并行都需要用户手动编程进行人工预分配, 而基于DALiuGE执行框架的并行处理方式可以实现负载均衡的自动调度, 无需用户进行额外的人工调度;

(2)在管线的并行执行与控制方面, 基于任务排队的Bash脚本并行方式执行过程缺乏实时监测和控制; MPI并行方式虽可以通过编程实现执行过程的故障检测, 但故障检测的延迟性和后续恢复的成本仍非常高. 这两种方式是由任务处理来驱动的简单并行流程, 无法自动根据任何中间数据集和刚好可用的资源来动态地调整任务执行, 往往需要更多的人工干预, 这对于SKA规模(数千万个并发任务)的数据处理工作流程来说是不现实的. 而基于DALiuGE执行框架的并行执行是由数据驱动的, 它将数据建模为逻辑图的节点, 在内存中实现为活动对象, 可以持续监控并触发后续任务的执行, 同时其采用去中心化方式执行任务节点和数据节点, 在硬件条件满足的情况下, 可以并发管理和执行数万个任务, 极大地提升了管线的并行可扩展性和灵活性. 此外, 执行框架中还集成了一个数据生命周期管理组件, 可以跟踪节点活动并在必要时自动迁移或删除;

(3)在管线的易用性和可迁移性方面, 前两种方式执行管线都需要用户具备专业编程知识, 通过脚本文件编程修改管线的运行设置来完成本地化部署, 部署步骤繁琐且使用过程不够简单清晰. 而使用DALiuGE执行框架的并行方式运行管线, 由于我们已将管线步骤集成入执行框架中生成系统组件, 用户只需在逻辑图编辑器EAGLE中导入已制作好的成像管线逻辑图, 即可通过修改步骤组件的可视化设置选项来实现无需编程的管线部署运行, 用户可以直观地调整管线的各个步骤和并行设置, 极大地增强了管线的易用性和可迁移性.

6.2　性能测试

实验环境由CSRC-P原型机的计算集群上的23个Intel X86计算节点和额外的管理与登录节点以及4.5 PB的分布式存储系统组成. 其中, 计算节点包括8个Purley CPU节点和15个华为CPU节点, 节点详细参数指标如表3所示. 分布式存储系统采用分层的存储设计, 使用全闪存NVMe存储以及SSD固态硬盘作为数据交换介质, 总存储为4.5 PB

(Huawei oceanstor). 为了比较性能, 我们使用相同的16个数据和运行参数在相同的环境中测试, 分别使用基于任务排队系统的管线Bash脚本并行方式、基于MPI的并行扩展方式和基于DALiuGE执行框架的自动化并行处理方式运行GLEAM-X成像管线. 具体运行参数如下: 并行使用计算节点数为16个, 每个节点使用内存为500 GB, 每个节点使用核心数为24、进程数为48. 根据我们的实验, 这3种并行方式的输入和输出结果完全相同, 这证明了我们在DALiuGE上的移植管线与原管线的功能等效.

6.3 结果分析

根据实验结果可以看出, DALiuGE表现出比MPI和Bash版本更好的性能. 图6显示了用3种方式处理1、2、4、8、16个快照数据(每个快照数据处理占据1个计算节点)时的实验结果.

表 3 计算节点主要参数指标

Table 3 Main parameters and indexes of compute node

Node	CPU (cores)	Memory (capacity)
Management and login	2×E5-4610v4 (12)	8×32 G DDR4 (256 GB)
Purley	2×intel (R) Gold6132 (28)	16×64 G DDR4 (1 TB)
Huawei	2×intel (R) Gold5218 (32)	12×64 G DDR4 (768 GB)

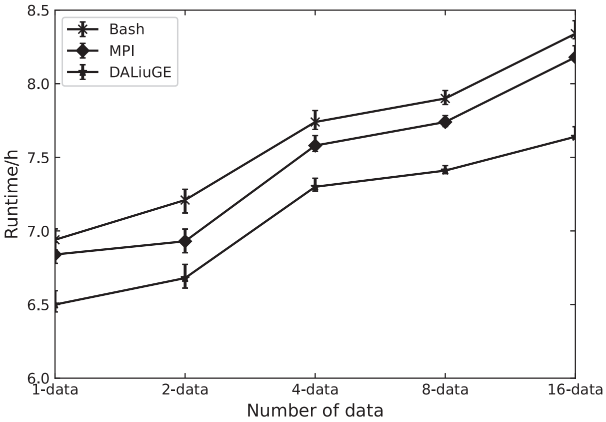

图 6 基于Bash、MPI和DALiuGE的管线的性能对比

Fig. 6 Performance comparison of pipelines based on Bash, MPI and DALiuGE

DALiuGE的总平均处理时间为7.1 h, 而MPI和Bash分别为7.48 h和7.67 h, 图中误差棒(标准差)为测试每种并行方式在相同并行数据数量下实验10次得出的, 可以看出在相同数量数据的情况下, DALiuGE的处理时间始终比MPI、Bash方式要少0.5 h到1 h, 且DALiuGE误差棒较短、多次处理时间的标准差较小, 因此DALiuGE的处理运行更为稳定且耗时更少. 随着并行处理数据数量的增加, 处理时间随之线性增加, 但DALiuGE依然是3种并行方式中处理时间最少的, 并且在并行扩展时, DALiuGE能在部署执行时为多条并行管线自动调度分配计算节点; 而无论是MPI还是Bash并行方式都需要用户先查询计算资源的空闲情况, 然后再通过额外编程来手动对多条并行管线进行计算节点的分配, 这将会产生更多的人工调度时间成本, 并且随着并行扩展数量的增加, 人工调度的时间成本也将越来越大, 这对于需要自动化处理海量数据的成像管线来说是很大的缺陷. 因此, 可以得出结论, 在以上3种并行方式中DALiuGE并行处理的GLEAM-X成像管线综合性能最好.

7 总结与展望

作为目前建设中最大的射电望远镜, SKA不仅有着孕育世界级科研成果的使命, 而且将会产生世界上最大规模的数据, 因此我们需要充分认识到SKA数据处理的巨大挑战[34]. 从技术趋势来看, 使用执行框架来处理SKA的超海量数据, 很可能是必然的选择, DALiuGE正是在这种背景下被提出和开发的[33].

本文基于SKA-1先导望远镜数据研究实现自

动化分布式数据处理的成像管线, 通过分析SKA-1分布式数据工作流, 研究管线与分布式工作流的DALiuGE执行框架集成方法、任务分配和负载均衡方法, 实现了具有较高可扩展性的分布式数据处理管线. 根据我们的实验, 集成入DALiuGE执行框架的成像管线在运行性能上对比传统的并行处理方式具备一定优势, 主要在于快速的任务定制和面向数据驱动的处理模型, 这符合科学区域中心的应用需求. 基于DALiuGE执行框架进行优化的成像管线相较于一般并行处理方式具有更好的性能优势、更灵活的适配性和可扩展性, 可支持未来SKA第一阶段试运行期间的大规模连续谱成像实验. 本文主要研究管线在并行框架上的整体优化, 注重于管线在多个计算节点上的并行度和可拓展性的提升以及管线在部署运行时的便利性, 用以解决大规模科学数据自动化处理问题, 对于管线内部步骤的执行代码不做更细粒度的并行优化. 对管线的内部细节优化以提升管线在单节点上的运行效率并与当前在并行框架上的整体性优化相结合将作为我们接下来更进一步的研究课题. 此外, DALiuGE的移植工作相当耗费人力, 实际上, 将成像管线移植到DALiuGE上是一项相当艰苦的工作, 这项工作在技术上的实现并不困难, 但却要消耗大量的人力. 在本次研究中, 由于没有辅助工具, DALiuGE的代码调试效率很低, 我们花费了大量的时间和精力在移植和新代码设计中, 因此在未来如何合理安排开发计划和开发人员来完成最终的DALiuGE的移植开发仍是一个需要研究的问题.

致谢 论文使用了GLEAM-X管线开发团队的代码, 论文作者感谢开发团队所有成员包括Natasha Hurley-Walker, Paul Hancock, Gemma Anderson, John Morgan, Stefan Duchesne和Tim Galvin. (github链接: https://github.com/tjgalvin/GLEAM-X-pipeline)

参考文献

[1] Weltman A, Bull P, Camera S, et al. PASA, 2020, 37: e002
[2] Scaife A M M. RSPTA, 2020, 378: 20190060
[3] Chrysostomou A, Taljaard C, Bolton R, et al. Observatory Operations: Strategies, Processes, and Systems VIII. SPIE: IEEE, 2020, 11449: 156
[4] van Haarlem M P, Wise M W, Gunst A W, et al. A&A, 2013, 556: A2
[5] Muxlow T W B, Thomson A P, Radcliffe J F, et al. MNRAS, 2020, 495: 1188
[6] Johnston S, Taylor R, Bailes M, et al. ExA, 2008, 22: 151
[7] Lonsdale C J, Cappallo R J, Morales M F, et al. IEEEP, 2009, 97: 1497
[8] Tingay S J, Goeke R, Bowman J D, et al. PASA, 2013, 30: e007
[9] Wayth R B, Lenc E, Bell M E, et al. PASA, 2015, 32: e025
[10] Hurley-Walker N, Galvin T J, Duchesne S W, et al. PASA, 2022, 39: e035
[11] Wu C, Tobar R, Vinsen K, et al. A&C, 2017, 20: 1
[12] Dean J, Ghemawat S. Communications of the ACM, 2008, 51: 107
[13] Isard M, Budiu M, Yu Y, et al. Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007. Lisbon Portugal. New York, NY, USA: ACM, 2007: 59
[14] Armbrust M, Xin R S, Lian C, et al. Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. Melbourne: IEEE, 2015: 1383
[15] Sokolowski M, Jordan C H, Sleap G, et al. PASA, 2020, 37: e021
[16] Offringa A R, Wayth R B, Hurley-Walker N, et al. PASA, 2015, 32: e008
[17] Offringa A R, van de Gronde J J, Roerdink J B T M. A&A, 2012, 539: A95
[18] Team C. ascl soft, 2019, 12: 002
[19] van Diepen G N J. A&C, 2015, 12: 174
[20] Offringa A R, Trott C M, Hurley-Walker N, et al. MNRAS, 2016, 458: 1057
[21] Offringa A R, McKinley B, Hurley-Walker N, et al. MNRAS, 2014, 444: 606
[22] Offringa A R, Smirnov O. MNRAS, 2017, 471: 301
[23] Hurley-Walker N, Callingham J R, Hancock P J, et al. MNRAS, 2017, 464: 1146
[24] Rich J W, de Blok W J G, Cornwell T J, et al. AJ, 2008, 136: 2897
[25] Hancock P J, Trott C M, Hurley-Walker N. PASA, 2018, 35: e011
[26] Hurley-Walker N, Filipović M D, Gaensler B M, et al. PASA, 2019, 36: e045
[27] Hurley-Walker N, Hancock P J. A&C, 2018, 25: 94
[28] Duchesne S W, Johnston-Hollitt M, Zhu Z, et al. PASA, 2020, 37: e037

[29] Sokolowski M, Colegate T, Sutinjo A T, et al. PASA, 2017, 34: e062

[30] Kurtzer G M, Sochat V, Bauer M W. PLoS One, 2017, 12: e0177459

[31] 劳保强, 安涛. 中国科学: 物理学 力学 天文学, 2020, 50: 135

[32] Taylor M B. ADASS, 2005, 347: 29

[33] Mei Y, Wei S, Wang F, et al. A&C, 2022, 38: 100541

[34] 安涛, 武向平, 洪晓瑜, 等. 中国科学院院刊, 2018, 33: 871

Low-frequency Radio Interferometric Array Imaging Pipeline Optimization Based on Distributed Execution Framework

WEI Yao-jie[1,2] FU Jie-lin[1] LAO Bao-qiang[2,3]

(1 Key Laboratory of Cognitive Radio and Information Processing of the Ministry of Education, School of Information and Communication, Guilin University of Electronic Science and Technology, Guilin 541004)
(2 Shanghai Astronomical Observatory, Chinese Academy of Sciences, Shanghai 200030)
(3 School of Physics and Astronomy, Yunnan University, Kunming 650500)

Abstract The Square Kilometre Array (SKA) project is an international collaboration to build the world's largest radio telescope, whose sensitivity and measurement speed will be an order of magnitude higher than those of all current radio telescopes. Radio continuum survey is one of the main observation mode of the SKA, and the establishment of a standard map of the survey area based on continuum imaging will provide an important foundation for subsequent astronomical science. The GaLactic and Extragalactic All-sky Murchison Widefield Array survey eXtended (GLEAM-X) is a project of the SKA pilot telescope Murchison Widefield Array (MWA) in 2018—2020. GLEAM-X is a new radio continuum survey project to be carried out with the MWA Phase II expansion array in 2018—2020. The experience of optimizing the imaging pipeline based on the distributed execution framework will help to solve the problem of massive data processing. In this paper, we describe the process steps of GLEAM-X imaging pipeline, integrate and improve it, and realize parallel processing of multiple pipelines on the China SKA Regional Centre Prototype (CSRC-P), and verify the deployment and test the correctness of the imaging pipeline system using GLEAM-X observation data. The GLEAM-X observations were used to validate the deployment of the imaging pipeline system and test its correctness. Then, to optimize the pipelines and improve the processing efficiency, the Data Activated Liu Graph Engine (DALiuGE) was used to integrate the imaging pipelines into the DALiuGE execution framework to automate the distributed parallel processing of the pipelines. Performance tests and results analysis show that the optimized imaging pipeline based on the DALiuGE execution framework has better performance, more flexible adaptability and scalability than the traditional parallel approach, and can support future large-scale continuum imaging experiments during the first phase of SKA commissioning.

Key words instrumentation: interferometers, methods: observational, techniques: interferometric, techniques: image processing, radio continuum: general